Jan 25, 2019 Version 1

Protocol for a reproducible circRNA analysis using Docker4Circ V.1

DOI
dx.doi.org/10.17504/protocols.io.wwaffae

Giulio Ferrero[1], Nicola Licheri[1], Lucia Coscujuela Tarrero[1], Carlo De Intinis[1], Valentina Miano[1], Raffaele Adolfo Calogero[1], Francesca Cordero[1], Marco Beccuti[1], Michele De Bortoli[1]

[1]University of Turin

Q-Bio Turin

Giulio Ferrero
University of Turin

DOI: **dx.doi.org/10.17504/protocols.io.wwaffae**

Protocol Citation: Giulio Ferrero, Nicola Licheri, Lucia Coscujuela Tarrero, Carlo De Intinis, Valentina Miano, Raffaele Adolfo Calogero, Francesca Cordero, Marco Beccuti, Michele De Bortoli 2019. Protocol for a reproducible circRNA analysis using Docker4Circ. **protocols.io** **https://dx.doi.org/10.17504/protocols.io.wwaffae**

Manuscript citation:

Protocol status: Working
We use this protocol and it's working

Created: January 08, 2019

Last Modified: January 25, 2019

Protocol Integer ID: 19106

Keywords: circRNA, colorectal cancer, pipeline, reproducible analysis

Abstract

Despite many computational tools were developed to predict circular RNAs (circRNAs), a limited number of work-flows exists to fully analyse a circRNA set ensuring the computational reproducibility of the whole analysis.
For this purpose, we designed Docker4Circ, a computational work-flow for a comprehensive circRNA analysis of a circRNAs composed of four modules: the circRNA prediction (module 1), the circRNA classification and annotation (module 2), the circRNA sequence analysis (module 3), and circRNA expression analysis (module 4).
To ensure reproducibility each function of Docker4Circ was embeded into a docker image following guideline provided by Reproducible Bioinformatics Project (RBP, **http://reproducible-bioinformatics.org/)**. Each function is included in the Docker4Seq R package which already includes different solutions for reproducible bioinformatic analyses.
This protocol describes the use of each function of Docker4Circ to analyse the circRNAs predicted from a set of RNA-Seq experiments performed in normal colon and colorectal cancer cell lines. Furthermore, the description of the functions required to compute the expression level of the analysed circRNAs in a set of RNA-Seq experiments of colorectal primary tumors is provided.

Before start

Before starting the analysis, the softwares R and Docker must be installed.

To install R follow the information reported in:
https://www.r-project.org/

To install docker follow the information reported in:
https://docs.docker.com/engine/installation/.

The pipeline is expected to run on 64 bits linux machine with at least 4 cores. 32 Gb RAM are required only if mapping will be done with STAR. A scratch folder should be present, e.g. /data/scratch and it should be writable from everybody.

Each Docker4Seq function has its specific set of parameters that are fully described in the documentation of each function. Access to the documentation using the "?" R operator followed by the name of the function (?ciri).

Softwares and datasets preparation

1 **Installation of the docker4seq R package.**

In the R environment digit the following command to install and run the docker4seq pipeline

Command

docker4seq R package installation

```
# Installation
install.packages("devtools")
library("devtools")
install_github("kendomaniac/docker4seq", ref="devel")

# Program execution
library("docker4Seq")
```

2 **Creation of the analysis folders.**

Before running the analyses, create the following directories in the working folder "**data**" "**reference**" "**scratch**".

The "data" folder will be used to store all the input and output data, the "scratch" folder will be used to store the temporary data, and the "reference" folder will be used to store the reference genome and annotation data.

The folders can be created in your working directory using the command "**mkdir**".

Command

Creation of working folders

```
# Creation of the directories for the analysis
system("mkdir data reference scratch")
```

Based on the path of your working directory, the path to these folder will be different. In our example the working directory is "**/home"**. Then, the path to these three folders will be "**/home/data**" "**/home/scratch**" "**/home/reference**"

You can obtain the path to your working directory using the R getwd() command.

3 **Download of the input RNA-Seq data**
The RNA-Seq raw reads for the circRNA prediction can be downloaded from **PRJNA393626**. This dataset includes a triplicate paired-end total and RNAse-R treated RNA-Seq performed on **NCM460** (normal colon cells), **SW480** (primary colorectal cancer cells), and **SW620** cell lines (metastatic colorectal cancer cells).

Dataset

PRJNA411984 NAME

https://www.ebi.ac.uk/ena/data/view/PRJNA411984 LINK

The detailed list of datasets needed for the analysis is the following:

SRA ID	**ID**
SRR5889371	NCM460_rep1
SRR5889366	NCM460_rep2
SRR5889367	NCM460_rep3
SRR5889368	SW480_rep1
SRR5889369	SW480_rep2
SRR5889362	SW480_rep3
SRR5889363	SW620_rep1
SRR5889364	SW620_rep2

SRR5889365 SW620_rep3

The the .fastq files can be downloaded from the ENA page of the experiment (linked below). The links to these files are reported under the column "**FASTQ files (FTP)**". Each dataset should be located in a specific directory, for example the experiment SRR5889371 should be put in the folder /data/SRR5889371.

Use the following code to automatically create these folders:

Command

Creation of the dataset folders

```
# Define a list containing the dataset ids
ids = c("SRR5889362", "SRR5889363", "SRR5889364", "SRR5889365",
"SRR5889366", "SRR5889367", "SRR5889368", "SRR5889369", "SRR5889371")

# Define a variable storing the path to data folder
data = paste0(getwd(), "/data")

# Create each dataset folder
for(i in 1:length(ids)){

dir.create(paste0(data, "/", ids[i]))

}
```

To download the files you can use the following R code.

Command

Dataset download

```
# Define a variable storing the fastq url
url1 <-
"ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR588/002/SRR5889362/SRR5889362_1
.fastq.gz"
url2 <-
"ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR588/002/SRR5889362/SRR5889362_2
.fastq.gz"

# Call wget to download the dataset into a specific folder
system(paste("wget", url1, "-P", paste0(data, "/SRR5889362")))
system(paste("wget", url2, "-P", paste0(data, "/SRR5889362")))
```

Module 1a: circRNA prediction with CIRI2

4 **Indexing of the reference genome**

The first module of Docker4Circ was designed for the prediction of circRNAs from raw RNA-Seq reads. In this part of the protocol, the prediction performed using the **CIRI2** algorithm is described. The first step of the pipeline is the BWA indexing of a reference genome using the ***bwaIndex*** function.

Command

Human genome BWA indexing

```
# Define a variable storing the path to the reference folder
ref <- paste0(getwd(), "/reference")

# Define a variable storing the url to the reference genome sequence
genome <-
"ftp://ftp.ensembl.org/pub/grch37/update/fasta/homo_sapiens/dna/Homo_sapiens.GRCh37.dna.toplevel.fa.gz"

# BWA indexing of the genomic sequence
bwaIndex(group="docker", genome.folder=ref, genome.url=genome,
mode="General")
```

All the index files of the reference genome will be stored in the reference folder.

5 **Read alignment with BWA**

The second step of the pipeline will be the RNA-Seq read quality control using the ***fastqc*** function, the read aligment using the ***bwa*** function, and the circRNA prediction using ***ciri2*** function. The sequential call of these three functions is implemented in the ***wrapperCiri*** function that can be used as follow:

Command

BWA RNA-Seq read alignment

```
# Define a variable storing the path to the scratch folder
scratch = paste0(getwd, "/scratch")

# Define a variable storing the path to the reference genome sequence
gs <- paste0(reference, "/Homo_sapiens.GRCh37.dna.toplevel.fa")

# Download the reference gene annotations
url_gene <-
"ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_28/GR
Ch37_mapping/gencode.v28lift37.annotation.gtf.gz"

system(paste("wget", url_gene, "-F", ref))

# Define a variable storing the path to the reference gene annotations
gtf <- paste0(reference, "/gencode.v28lift37.annotation.gtf.gz")

# Repeat the BWA alignment for each dataset
for (i in 1:length(ids)){

fqfolder <- paste(data, ids[i] , sep="/")

wrapperCiri(group = "docker", scratch.folder=scratch,
data.folder=fqfolder,
genome.file=gs, seq.type="pe", sample.id=ids[i], threads=1,
annotation.file=gtf,
max.span=200000, stringency.value="high", quality.threshold=10)

}
```

The read alignment quality control files, the BWA read alignment files and the set of predicted circRNAs (.ciri files) will be stored in the folder in which the input fastq files are located.

6 **Merge of multiple CIRI2 predictions**

To merge multiple CIRI2 circRNA predictions into a common list of circRNA, the function **ciri2MergePredictions** was implemented. Each .ciri files located in a specific dataset folder will be merged into a single file. Each sample name should be reported ad element of the **samples.list** vector with correponding sample type specified as element of the **covariate.list** vector.

Command

Merge of different CIRI2 predictions

```
# Copy each CIRI2 prediction in the data folder
system(paste0("cp ", data, "/*/*.ciri ", data))

# Define a list of sample groups
cov <- c("1","1","1","2","2","2","3","3","3")

# Merge the CIRI2 outputs
ciri2MergePredictions(group="docker", scratch.folder=scratch,
data.folder=data,
 samples.list=ids, covariate.list=cov, min_reads=2, min_reps=2,
min_avg=10)
```

This function generates two files named **merged_circs.crna** and **merged_circs.crna_count**. The first file reports the ids and coordinates of each circRNA while the second file is a matrix reporting the number of backsplicing reads supporting each circRNA in each dataset.

Module 1b: circRNA prediction with STARChip

7 **Indexing of the reference genome**

In this part of the protocol, the prediction performed using the **STARChip** algorithm is described. The first step of the pipeline is the STAR indexing of a reference genome using the ***rsemstarIndex*** function.

Command

Reference genome indexing with STAR

```
# Define a variable storing the url of the gene annotations
url_gene <-
"tp://ftp.ensembl.org/pub/grch37/update/gtf/homo_sapiens/Homo_sapiens.
GRCh37.87.gtf.gz"

# Call the function to index the reference genome
rsemstarIndex(group="docker",  genome.folder=ref,
ensembl.urlgenome=genome, ensembl.urlgtf=url_gene, threads=40)
```

The STAR index files of the reference genome will be stored in the **reference** folder.

8 **Create the reference file for STARChip**

The STARChip algorithm required an additional .bed file defined starting from the STAR-indexed reference genome. To generate this file, the function ***starChipIndex*** should be applied as follow:

Command

Reference creation for STARChip

```
# Creation of the STARChip reference file
starChipIndex(group="docker", genome.folder=ref)
```

Two files called **genome.gtf.bed** and **genome.gtf.exon.bed** will be generated in the **reference** folder.

9 **Running of STAR chimeric analysis**

The STARChip circRNA prediction require a run of STAR chimeric algorithm on the RNA-Seq reads. The STAR chimeric analysis is implemented in the ***starChimeric*** function that shoul be applied as follow:

Command

STAR chimeric analysis

```
# Run of the STAR chimeric analysis on each RNA-Seq dataset
for(i in 1:length(ids)){

fqfolder = paste(path, ids[i] , sep="")

starChimeric(group="docker", fastq.folder=fqfolder,
scratch.folder=scratch, genome.folder=ref threads=40,
chimSegmentMin=20, chimJunctionOverhangMin=15)

}
```

This function will generate each STAR chimeric output in each sample directory.

10 **STARChip circRNA prediction**
The circRNA prediction will be performed using the ***starchipCircle*** function applied on the **data** folder containing each sample-specific folder

Command

STARChip circRNA prediction

```
# Prediction of circRNAs using STARChip
starchipCircle_temp(group="docker", genome.folder=ref,
scratch.folder=scratch,
samples.folder=data, reads.cutoff=1, min.subject.limit=2, threads=40,
do.splice="True", cpm.cutoff=0, subjectCPM.cutoff=0,
annotation="true")
```

Module 2: circRNA annotation and classification

11 **Preparation of the reference files for the classification**
To classify the circRNAs a reference of Ensembl exon and transcript annotations is required. To provide these data in the correct format, the ***circrnaPrepareFiles*** function should be applied as follow.

Command

Preparation of the reference files for the classification

```
# Create the reference exon and gene isoforms files required for the
circRNA classification analysis
circrnaPrepareFiles(group="docker", scratch.folder=scratch,
data.folder=data,
assembly="hg19")
```

Two files called **exons_reference** and **isoforms_reference** reporting the exons and the gene isoform annotations respectively, will be generated in the data folder.

12 **Classification of circRNAs based on their mapping location**
To classify the circRNAs into distinct classes the function ***circrnaClassification*** should be applied as follow.

Command

circRNA classification

```
# Classification of a circRNA set
circrnaClassification(group="docker", scratch.folder=scratch,
circrna.data=paste0(data,"/merged_circs.crna"),
exon.data=paste0(data,"/exons_reference",
isoform.data=paste0(data,"/isoforms_reference"), assembly="hg19"))
```

Two files called **circRNA_classification** and **circRNA_univocal_classification** reporting the transcript- and the gene-level classification respectively, will be generated in the data folder.

13 **Annotation of the circRNA based on databases information**
To annotate the circRNAs based on the information stored in the circBase and TSCD database, the function ***circAnnotation*** should be applied as follow.

Command

circRNA annotation

```
# Annotation of a circRNA set based on information from public
databases
circAnnotations(group="docker", scratch.folder=scratch,
 ciri.file=paste0(data,"/merged_circs.crna"), genome.version="hg19")
```

Different files with suffix **.anno** will be will be generated in the data folder. Each of these files will report the information about the circRNAs found in a specific database (es. circbase.anno will report the circBase information)

Module 3: circRNA sequence analysis

14 **Reconstruction of the circRNA backsplicing junction sequences**
To reconstruct the sequence of the backsplicing junction of each circRNA the function ***circrnaBSJunctions*** should be applied as follow

Command

Backsplicing sequence reconstruction

```
# Reconstruction of circRNA backsplicing junction sequence
circrnaBSJunctions(group="docker", scratch.folder=scratch,
circrna.data=paste0(data,"/merged_circs.crna"),
exon.data=paste0(data,"/exons_reference"),
assembly="hg19")
```

This function will generate a files called **circRNA_backsplicing_sequences.fasta** reporting the reconstructed sequence.

15 **Prediction of alternative splicing events involving the circRNAs**

To predict the set of alternative splicing events involving the circRNA, the ***ciriAS*** function can be applied as follow using each BWA read alignment file and CIRI2 output.

Command

Detection of alternative splicing events involving the circRNAs

```
# Application of CIRI AS algorithm on each dataset
for(i in 1:length(ids)){

fqfolder = paste(path, ids[i] , sep="")

ciriAS(group="docker", scratch.folder=scratch,
sam.file=paste0(fqfolder,"/aligned_reads.sam"),
ciri.file=paste0(fqfolder, "/", ids[i], ".ciri"), genome.file=genome,
annotation.file=gtf)

}
```

The output of the function will be a two different files called **structure.list** and **structure_AS.list**.

Module 4: circRNA expression analysis

16 **Download of the RNA-Seq data in which quantify the expression of a circRNA set**

The RNA-Seq raw reads for the circRNA quantification can be downloaded from **PRJNA-411984**. This dataset includes RNA-Seq data of three CRC samples with matched data from normal mucosa samples.

Dataset

PRJNA411984 NAME

https://www.ebi.ac.uk/ena/data/view/PRJNA411984 LINK

The detailed list of datasets needed for the analysis is the following:

SRA ID	ID
SRR6067723	CRC_1
SRR6067725	CRC_2
SRR6067727	CRC_3
SRR6067729	Normal mucosa 1
SRR6067731	Normal mucosa 2
SRR6067733	Normal mucosa 3

For each dataset a specific folder in the data folder

The datasets can be downloaded using the following code applied using the url of the different datasets.

Command

Download dataset

```
# Define a variable storing the fastq url
url1 <-
"ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR606/003/SRR6067723/SRR6067723_1
.fastq.gz"
url2 <-
"ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR606/003/SRR6067723/SRR6067723_2
.fastq.gz"

# Call wget to download the dataset into a specific folder
system(paste("wget", url1, "-P", paste0(data, "/SRR6067723")))
system(paste("wget", url2, "-P", paste0(data, "/SRR6067723")))
```

17 **Quantification of circRNA expression in RNA-Seq datasets**

The quantification of each circRNA in a specific RNA-Seq dataset is computed using the ***circrnaQuantification*** function

Command

CircRNA expression analysis

```
# Define a list containing the dataset ids
ids2 <- c("SRR6067723", "SRR6067725", "SRR6067727", "SRR6067729",
"SRR6067731", "SRR6067733")

# Apply the quantification function on the fastq file stored in each
sample folder
for(i in 1:length(ids2)){

fqfolder = paste0(data, "/", ids2[i] , sep="")

setwd(fqfolder)

# For paired-end reads, the two fastq files must be joined into a
single fastq file
system(paste0("cat ", ids2[i], "_1.fastq ",  ids2[i], "_2.fastq > ",
ids2[i], ".fastq"))

circrnaQuantification(group="docker", scratch.folder=scratch,
rnaseq.data=paste0(fqfolder, "/", ids2[i], ".fastq"),
backsplicing_junctions.data=paste0(data,
"/circRNA_backsplicing_sequences.fasta"),
hc.params=c(21, 40, 1000000, 1000000, 17, 30))

}
```

This function will generate a file with extension **.hashcirc** reporting for each dataset the number of reads supporting each circRNA.

18 **CircRNA differential expression analysis**

Before the differential expression analysis of the circRNA level detected by the circrnaQuantification function, the function ***mergeData*** should be applied in order to merge the result obtained for each dataset.

Command

Merge different data

```
# Define a variable storing the list of sample classes
cov <- c("T", "T", "T", "N", "N", "N")
cov.order <- c("T", "N")

# Merge different file with extension hashcirc in the subfolders of
data
mergeData(group="docker", scratch.folder=scratch, sample.ids=ids2,
covariates=cov, covariate.order=cov.order, data.folder=data,
extension="hashcirc", column_index=2)
```

The output of this function called **MergedData.tsv** is a table reporting the counts of each annotation for each dataset. Furthermore, the sample identifiers reported in the header will be report the sample id followed by the corresponding sample class. On this table the differential expression analysis can be perfomed using DESeq2 algorithm implemented in the ***wrapperDeseq2*** function.

Command

Differential expression analysis

```
# Differential expression analysis with DESeq2
wrapperDeseq2(group="docker",  output.folder=data,
experiment.table=paste0(data, "/MergedData.tsv"), log2fc=0, fdr=0.05,
ref.covar="N",
type="gene", batch=FALSE)
```